UNIVERSITÀ DEGLI STUDI DI MESSINA
Dipartimento di Ingegneria

Corso di Laurea Magistrale in

Engineering and Computer Science (LM-18

LM-32)

Subject: Computer System Analysis

System with concurrent jobs

Student:
Abdirashova Aruzhan

Docente:
Prof. Marco Lucio Scarpa

A.A. 2024

Exercise

My study focuses on the performance evaluation of a service system handling two types of requests with varying arrival and service rates. The system operates with maximum capacities for each request type and employs either random or priority-based queue selection methods.
System Parameters:

- Maximum capacity for type one requests (N1): 8
- Maximum capacity for type two requests (N2): 12
- Arrival rate for type one requests (λ1): 2 requests/min
- Arrival rate for type two requests (λ2): 1.25 requests/min
- Service rate for type one requests (μ1): 3.33 requests/min
- Service rate for type two requests (μ2): 0.67 requests/min

We need to evaluate performance metrics such as system throughput, loss rates, server utilization, mean total number of requests, and mean number of requests by type.

We compare system performance under two scenarios:

- Random Queue Selection: Requests are served from both queues randomly.
- Priority-based Queue Selection: Requests are served based on priority, favoring one queue over the other.

Experiment

First of all, the behavior of complex systems can be easily and efficiently represented by using GSPN, rather than using Markov chains directly. The M/M/1//(N1,N2) queue has been observed and toward it have been built a Petri network. I have used the PN Toolbox for Matlab for my simulation. Typical for these types of queues is that results are calculated with considering the execution (service) time.

Case 1 - The Figure 1 shows the petri network for random queue before performed simulation for costumers. For each customer class we use a place for representing the customers in queue(P.Queue), and a place for the customer in service(P.Serv). Source place stores as many tokens as its maximum N1 or N2 requests. If server is free and a set of waiting customers in immediate transition puts a customer in service. Here we choose the customer to put in service probabilistically, according to the uniform distribution. This is obtained by an appropriate definition of the immediate transition weights. The firing of T.arr1 is marking dependent. It means that the firing rate of T.arr1 depends on the number of tokens in P.Sour1. If we have N1 tokens in P.Sour1, the firing rate is N1λ1.

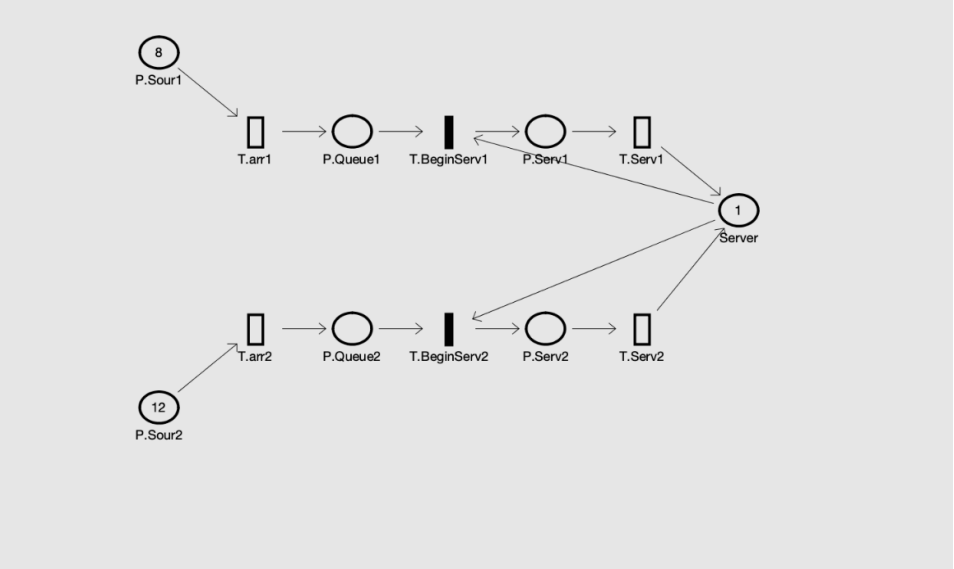

Figure-1: GSPN representation for random queue selection

Time: 24.6938
Events: 60

Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
P.Sour1	0	0	Inf	8	0.32397	3.0867	0.50119	0.16237
P.Sour2	0	0	Inf	12	0.48595	2.0578	0.72222	0.35097
P.Queue1	8	0.32397	3.0867	8	0.32397	3.0867	7.4573	2.4159
P.Queue2	12	0.48595	2.0578	12	0.48595	2.0578	11.0777	5.3832
P.Serv1	8	0.32397	3.0867	8	0.32397	3.0867	0.50286	0.16291
P.Serv2	12	0.48595	2.0578	12	0.48595	2.0578	1.7215	0.83655

Table 1: Statistics results: Places for Case 1

Transition Name	Service Sum	Service Rate	Service Dist.	Service Time	Utilization
T.arr1	8	0.32397	3.0867	0.083596	0.027082
T.arr2	12	0.48595	2.0578	0.073177	0.03556
T.BeginServ1	8	0.32397	3.0867	0	0
T.BeginServ2	12	0.48595	2.0578	0	0
T.Serv1	8	0.32397	3.0867	0.44994	0.14577
T.Serv2	12	0.48595	2.0578	1.6289	0.79159

Table 2: Statistics results: Transitions for Case 1

Case 2 - I consider priority queuing mechanism giving higher priority to the first queue(P.Sour1) over the second one(P.Sour2). If a type II customer finds the server idle upon arrival, it immediately goes for service. If the server is free at the time of arrival of a primary customer, then the customer starts to be served. Any high priority customer which, upon arrival, finds the server busy is queued up in an ordinary queue. Upon blocking, low priority customers immediately join a pool of unsatisfied customers, called the orbit. The immediate transition t.orbit fires at the arrival of a low priority customer who finds no operational free

server. Hence, it joins immediately the orbit represented by the place P.Orbit. Once in orbit, the customer starts generation of a flow of repeated calls exponentially distributed withwith rate ν, and is independent of all previous retrial times and all other stochastic processes in the system.

The immediate transition T.BeginServ2 fires if the place P.Queue1 is empty (This condition is expressed by the inhibitor arc from place P.Queue1 to the transition T.BeginServ2.), the place Server contains one token which represents the idle server and the place P.Queue2 contains one token. So, the place P.Serv2 receives a token representing a low priority customer in service.

The timed transition T.Serv2 (respectively, the timed transition T.Serv1) is fired to determine the end of the low priority ustomer period service (respectively, the high priority customer period service).

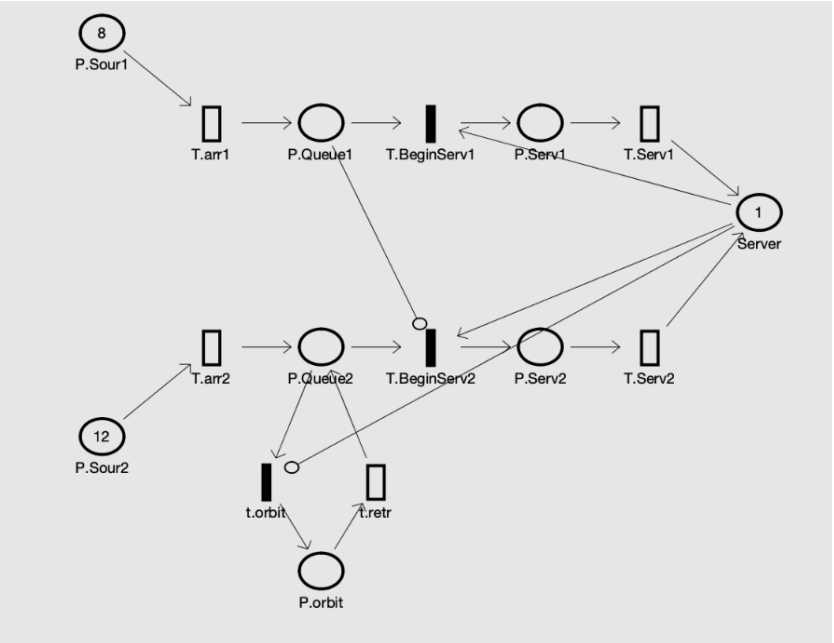

Figure-2: GSPN representation for priority queue selection

Time: 22.088
Event: 342

Place Name	Arrival Sum	Arrival Rate	Arrival Dist.	Throughput Sum	Throughput Rate	Throughput Dist.	Waiting Time	Queue Length
P.Sour1	0	0	Inf	8	0.36219	2.761	0.69532	0.25184
P.Sour2	0	0	Inf	12	0.54328	1.8407	0.43665	0.23722
P.Queue1	8	0.36219	2.761	8	0.36219	2.761	1.0238	0.3708
P.Queue2	153	6.9268	0.14437	153	6.9268	0.14437	0	0
P.Serv1	8	0.36219	2.761	8	0.36219	2.761	0.48951	0.17729
P.Serv2	12	0.54328	1.8407	12	0.54328	1.8407	1.3438	0.73008

Table 1: Statistics results: Places for Case 2

Transition Name	Service Sum	Service Rate	Service Dist.	Service Time	Utilization
T.arr1	8	0.36219	2.761	0.025767	0.0093324
T.arr2	12	0.54328	1.8407	0.036592	0.01988
T.BeginServ1	8	0.36219	2.761	0	0
T.BeginServ2	12	0.54328	1.8407	0	0
T.Serv1	8	0.36219	2.761	0.060296	0.021838
T.Serv2	12	0.54328	1.8407	0.28229	0.15336
t.retr	141	6.3836	0.15665	0.12463	0.79559

Table 2: Statistics results: Transitions for Case 2

Conclusion

Generalized Stochastic Petri nets have proven to be a powerful and enduring graphically oriented framework for modeling and performance analysing of complex systems. Finally, we conclude that the arrival rate of customers, retrial rate and the number of customers in the sources are the major factors affecting the performance of this system. From these results, we conclude that the Petri net theory can be used to add powerful analysis capabilities to high priority and retrial queueing systems.